Compilateur

Generated by Doxygen 1.8.11

Contents

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Data Structure Documentation

3.1 desc_identif Struct Reference

`#include <dico.h>`

Data Fields

- char * identif
- int classe
- int type
- int adresse
- int complement

3.1.1 Field Documentation

3.1.1.1 int desc_identif::adresse

3.1.1.2 int desc_identif::classe

3.1.1.3 int desc_identif::complement

3.1.1.4 char* desc_identif::identif

3.1.1.5 int desc_identif::type

The documentation for this struct was generated from the following file:

- inc/dico.h

3.2 dico_ Struct Reference

`#include <dico.h>`

Data Fields

- desc_identif tab [maxDico]
- int base
- int sommet

3.2.1 Field Documentation

3.2.1.1 int dico_::base

3.2.1.2 int dico_::sommet

3.2.1.3 desc_identif dico_::tab[maxDico]

The documentation for this struct was generated from the following file:

- inc/dico.h

3.3 n_appel_ Struct Reference

```
#include <syntabs.h>
```

Data Fields

- char * fonction
- n_l_exp * args

3.3.1 Field Documentation

3.3.1.1 n_l_exp* n_appel_::args

3.3.1.2 char* n_appel_::fonction

The documentation for this struct was generated from the following file:

- inc/syntabs.h

3.4 n_dec_ Struct Reference

```
#include <syntabs.h>
```

Public Types

- enum { foncDec, varDec, tabDec }

Data Fields

- enum n_dec_:: { ... } type
- char * nom
- union {
  struct {
  n_l_dec * param
  n_l_dec * variables
  n_instr * corps
  } foncDec_
  struct {
  int type
  } varDec_
  struct {
  int taille
  } tabDec_
  } u

3.4.1 Member Enumeration Documentation

3.4.1.1 anonymous enum

Enumerator

foncDec

varDec

tabDec

3.4.2 Field Documentation

3.4.2.1 n_instr* n_dec_::corps

3.4.2.2 struct { ... } n_dec_::foncDec_

3.4.2.3 char* n_dec_::nom

3.4.2.4 n_l_dec* n_dec_::param

3.4.2.5 struct { ... } n_dec_::tabDec_

3.4.2.6 int n_dec_::taille

3.4.2.7 enum { ... } n_dec_::type

3.4.2.8 int n_dec_::type

3.4.2.9 union { ... } n_dec_::u

3.4.2.10 struct { ... } n_dec_::varDec_

3.4.2.11 n_l_dec* n_dec_::variables

The documentation for this struct was generated from the following file:

- inc/syntabs.h

3.5 n_exp_ Struct Reference

`#include <syntabs.h>`

Public Types

- enum {
  varExp, opExp, intExp, appelExp,
  lireExp, incrExp }

Data Fields

- enum n_exp_:: { ... } type
- union {
  struct {
  operation op
  struct n_exp_ * op1
  struct n_exp_ * op2
  } opExp_
  n_var * var
  n_var * incr
  int entier
  n_appel * appel
  } u

3.5.1 Member Enumeration Documentation

3.5.1.1 anonymous enum

Enumerator

varExp

opExp

intExp

appelExp

lireExp

incrExp

3.5.2 Field Documentation

3.5.2.1 n_appel* n_exp_::appel

3.5.2.2 int n_exp_::entier

3.5.2.3 n_var* n_exp_::incr

3.5.2.4 operation n_exp_::op

3.5.2.5 struct n_exp_* n_exp_::op1

3.5.2.6 struct n_exp_* n_exp_::op2

3.5.2.7 struct { ... } n_exp_::opExp_

3.5.2.8 enum { ... } n_exp_::type

3.5.2.9 union { ... } n_exp_::u

3.5.2.10 n_var* n_exp_::var

The documentation for this struct was generated from the following file:

- inc/syntabs.h

3.6 n_instr_ Struct Reference

```
#include <syntabs.h>
```

Public Types

- enum {
  incrInst, affecteInst, siInst, faireInst,
  tantqueInst, appelInst, retourInst, ecrireInst,
  videInst, blocInst, pourInst }

Data Fields

- enum n_instr_:: { ... } type
- union {
  n_exp * incr
  struct {
    n_var * var
    n_exp * exp
  } affecte_
  struct {
    n_exp * test
    struct n_instr_ * alors
    struct n_instr_ * sinon
  } si_
  struct {
    n_exp * test
    struct n_instr_ * faire
  } tantque_
  struct {
    n_exp * test
    struct n_instr_ * faire
  } faire_
  struct {
    n_exp * test
    struct n_instr_ * init
    struct n_instr_ * incr
    struct n_instr_ * faire
  } pour_
  n_appel * appel
  struct {
    n_exp * expression
  } retour_
  struct {
    n_exp * expression
  } ecrire_
  n_l_instr * liste
  } u

3.6.1 Member Enumeration Documentation

3.6.1.1 anonymous enum

Enumerator

incrInst

affecteInst

siInst

faireInst

tantqueInst

appelInst

retourInst

ecrireInst

videInst

blocInst

pourInst

3.6.2 Field Documentation

3.6.2.1 struct { ... } n_instr_::affecte_

3.6.2.2 struct **n_instr_**∗ n_instr_::alors

3.6.2.3 **n_appel**∗ n_instr_::appel

3.6.2.4 struct { ... } n_instr_::ecrire_

3.6.2.5 **n_exp**∗ n_instr_::exp

3.6.2.6 **n_exp**∗ n_instr_::expression

3.6.2.7 struct **n_instr_**∗ n_instr_::faire

3.6.2.8 struct { ... } n_instr_::faire_

3.6.2.9 **n_exp**∗ n_instr_::incr

3.6.2.10 struct **n_instr_**∗ n_instr_::incr

3.6.2.11 struct **n_instr_**∗ n_instr_::init

3.6.2.12 **n_l_instr**∗ n_instr_::liste

3.6.2.13 struct { ... } n_instr_::pour_

3.6.2.14 struct { ... } n_instr_::retour_

3.6.2.15 struct { ... } n_instr_::si_

3.6.2.16 struct **n_instr_**∗ n_instr_::sinon

3.6.2.17 struct { ... } n_instr_::tantque_

3.6.2.18 **n_exp**∗ n_instr_::test

3.6.2.19 enum { ... } n_instr_::type

3.6.2.20 union { ... } n_instr_::u

3.6.2.21 **n_var**∗ n_instr_::var

The documentation for this struct was generated from the following file:

- inc/syntabs.h

3.7 n_l_dec_ Struct Reference

`#include <syntabs.h>`

Data Fields

- n_dec * tete
- struct n_l_dec_ * queue

3.7.1 Field Documentation

3.7.1.1 struct n_l_dec_* n_l_dec_::queue

3.7.1.2 n_dec* n_l_dec_::tete

The documentation for this struct was generated from the following file:

- inc/syntabs.h

3.8 n_l_exp_ Struct Reference

`#include <syntabs.h>`

Data Fields

- n_exp * tete
- struct n_l_exp_ * queue

3.8.1 Field Documentation

3.8.1.1 struct n_l_exp_* n_l_exp_::queue

3.8.1.2 n_exp* n_l_exp_::tete

The documentation for this struct was generated from the following file:

- inc/syntabs.h

3.9 n_l_instr_ Struct Reference

`#include <syntabs.h>`

Data Fields

- n_instr * tete
- struct n_l_instr_ * queue

3.9.1 Field Documentation

3.9.1.1 struct n_l_instr_* n_l_instr_::queue

3.9.1.2 n_instr* n_l_instr_::tete

The documentation for this struct was generated from the following file:

- inc/syntabs.h

3.10 n_prog_ Struct Reference

```
#include <syntabs.h>
```

Data Fields

- n_l_dec * variables
- n_l_dec * fonctions

3.10.1 Field Documentation

3.10.1.1 n_l_dec* n_prog_::fonctions

3.10.1.2 n_l_dec* n_prog_::variables

The documentation for this struct was generated from the following file:

- inc/syntabs.h

3.11 n_var_ Struct Reference

```
#include <syntabs.h>
```

Public Types

- enum { simple, indicee }

Data Fields

- enum n_var_:: { ... } type
- char * nom
- union {
  struct {
  n_exp * indice
  } indicee_
  } u

3.11.1 Member Enumeration Documentation

3.11.1.1 anonymous enum

Enumerator

simple

indicee

3.11.2 Field Documentation

3.11.2.1 n_exp* n_var_::indice

3.11.2.2 struct { ... } n_var_::indicee_

3.11.2.3 char* n_var_::nom

3.11.2.4 enum { ... } n_var_::type

3.11.2.5 union { ... } n_var_::u

The documentation for this struct was generated from the following file:

- inc/syntabs.h

Chapter 4

File Documentation

4.1 inc/affiche_arbre_abstrait.h File Reference

```
#include "syntabs.h"
```

Functions

- void affiche_n_prog (n_prog *n)

4.1.1 Function Documentation

4.1.1.1 void affiche_n_prog (n_prog * *n*)

4.2 inc/analyseur_lexical.h File Reference

```
#include "stdio.h"
```

Functions

- int yylex (void)
- void nom_token (int token, char *nom, char *valeur)
- void test_yylex_internal (FILE *yyin)

4.2.1 Function Documentation

4.2.1.1 void nom_token (int *token,* char * *nom,* char * *valeur*)

4.2.1.2 void test_yylex_internal (FILE * *yyin*)

4.2.1.3 int yylex (void)

4.3 inc/analyseur_semantique.h File Reference

```
#include "syntabs.h"
```

Functions

- void semantique (n_prog *p, int trace_dico, int print_mips)

  Main function to parse syntaxique tree.

4.3.1 Function Documentation

4.3.1.1 void semantique (n_prog * *p,* int *trace_dico,* int *print_mips*)

Main function to parse syntaxique tree.

Parameters

	p	The head of the tree
in	*trace_dico*	boolean to display symbol table
in	*print_mips*	boolean to display the mips output on stdin

4.4 inc/analyseur_syntaxique.h File Reference

```
#include "syntabs.h"
```

Functions

- n_l_dec * listeDecVariables (void)
- n_l_dec * listeDecFonctions (void)
- n_dec * declarationVariable (void)
- n_dec * declarationFonction (void)
- n_l_dec * listeParam (void)
- n_l_instr * listeInstructions (void)
- n_instr * instruction (void)
- n_instr * instructionAffect (void)
- n_instr * instructionBloc (void)
- n_instr * instructionSi (void)
- n_instr * instructionTantque (void)
- n_instr * instructionAppel (void)
- n_instr * instructionRetour (void)
- n_instr * instructionEcriture (void)
- n_instr * instructionVide (void)
- n_var * var (void)
- n_exp * expression (void)
- n_appel * appelFct (void)
- n_exp * conjonction (void)
- n_exp * negation (void)
- n_exp * comparaison (void)
- n_exp * expArith (void)
- n_exp * terme (void)
- n_exp * facteur (void)

- n_exp * argumentsEffectifs (void)
- n_l_exp * listeExpressions (void)
- n_l_exp * listeExpressionsBis (n_l_exp *herite)
- n_prog * programme (void)
- n_exp * conjonctionBis (n_exp *herite)
- int optTailleTableau (void)
- n_exp * expArithBis (n_exp *herite)
- n_instr * optSinon (void)
- n_exp * comparaisonBis (n_exp *herite)
- n_l_dec * optDecVariables (void)
- n_exp * optIndice (void)
- n_l_dec * listeDecVariablesBis ()
- n_exp * termeBis (n_exp *herite)
- n_exp * expressionBis (n_exp *herite)
- n_instr * instructionFaire (void)
- n_l_dec * optListeDecVariables (void)
- n_instr * instructionPour (void)
- void EatTerminal (void)
- void DisplayErreur (void)
- n_prog * syntaxe (int trace_xml_tree)

4.4.1 Function Documentation

4.4.1.1 n_appel* appelFct (void)

4.4.1.2 n_exp* argumentsEffectifs (void)

4.4.1.3 n_exp* comparaison (void)

4.4.1.4 n_exp* comparaisonBis (n_exp * *herite*)

4.4.1.5 n_exp* conjonction (void)

4.4.1.6 n_exp* conjonctionBis (n_exp * *herite*)

4.4.1.7 n_dec* declarationFonction (void)

4.4.1.8 n_dec* declarationVariable (void)

4.4.1.9 void DisplayErreur (void)

4.4.1.10 void EatTerminal (void)

4.4.1.11 n_exp* expArith (void)

4.4.1.12 n_exp* expArithBis (n_exp * *herite*)

4.4.1.13 n_exp* expression (void)

4.4.1.14 n_exp* expressionBis (n_exp * *herite*)

4.4.1.15 n_exp* facteur (void)

4.4.1.16 n_instr* instruction (void)

4.4.1.17 n_instr* instructionAffect (void)

4.4.1.18 n_instr* instructionAppel (void)

4.4.1.19 n_instr* instructionBloc (void)

4.4.1.20 n_instr* instructionEcriture (void)

4.4.1.21 n_instr* instructionFaire (void)

4.4.1.22 n_instr* instructionPour (void)

4.4.1.23 n_instr* instructionRetour (void)

4.4.1.24 n_instr* instructionSi (void)

4.4.1.25 n_instr* instructionTantque (void)

4.4.1.26 n_instr* instructionVide (void)

4.4.1.27 n_l_dec* listeDecFonctions (void)

4.4.1.28 n_l_dec* listeDecVariables (void)

4.4.1.29 n_l_dec* listeDecVariablesBis ( )

4.4.1.30 n_l_exp* listeExpressions (void)

4.4.1.31 n_l_exp* listeExpressionsBis (n_l_exp * *herite*)

4.4.1.32 n_l_instr* listeInstructions (void)

4.4.1.33 n_l_dec* listeParam (void)

4.4.1.34 n_exp* negation (void)

4.4.1.35 n_l_dec* optDecVariables (void)

4.4.1.36 n_exp* optIndice (void)

4.4.1.37 n_l_dec* optListeDecVariables (void)

4.4.1.38 n_instr* optSinon (void)

4.4.1.39 int optTailleTableau (void)

4.4.1.40 n_prog* programme (void)

4.4.1.41 n_prog* syntaxe (int *trace_xml_tree*)

4.4.1.42 n_exp* terme (void)

4.4.1.43 n_exp* termeBis (n_exp * *herite*)

4.4.1.44 n_var* var (void)

4.5 inc/dico.h File Reference

```
#include "syntabs.h"
```

Data Structures

- struct desc_identif
- struct dico_

Macros

- #define maxDico 100
- #define C_VARIABLE_GLOBALE 1
- #define C_VARIABLE_LOCALE 2
- #define C_ARGUMENT 3
- #define T_ENTIER 1
- #define T_TABLEAU_ENTIER 2
- #define T_FONCTION 3

Functions

- void entreeFonction (void)
- void sortieFonction (void)
- int ajouteIdentificateur (char *identif, int classe, int type, int adresse, int complement)
- int rechercheExecutable (char *identif)
- int rechercheDeclarative (char *identif)
- void affiche_dico (void)

Variables

- dico_ dico
- int contexte
- int adresseLocaleCourante
- int adresseArgumentCourant

4.5.1 Macro Definition Documentation

4.5.1.1 #define C_ARGUMENT 3

4.5.1.2 #define C_VARIABLE_GLOBALE 1

4.5.1.3 #define C_VARIABLE_LOCALE 2

4.5.1.4 #define maxDico 100

4.5.1.5 #define T_ENTIER 1

4.5.1.6 #define T_FONCTION 3

4.5.1.7 #define T_TABLEAU_ENTIER 2

4.5.2 Function Documentation

4.5.2.1 void affiche_dico (void)

4.5.2.2 int ajouteIdentificateur (char * *identif,* int *classe,* int *type,* int *adresse,* int *complement*)

4.5.2.3 void entreeFonction (void)

4.5.2.4 int rechercheDeclarative (char * *identif*)

4.5.2.5 int rechercheExecutable (char * *identif*)

4.5.2.6 void sortieFonction (void)

4.5.3 Variable Documentation

4.5.3.1 int adresseArgumentCourant

4.5.3.2 int adresseLocaleCourante

4.5.3.3 int contexte

4.5.3.4 dico_ dico

4.6 inc/freedom.h File Reference

```
#include "syntabs.h"
```

Macros

- #define __FREEDOM__ value

Functions

- void freedom (n_prog *p)

4.6.1 Macro Definition Documentation

4.6.1.1 #define __FREEDOM__ value

4.6.2 Function Documentation

4.6.2.1 void freedom (n_prog * *p*)

4.7 inc/premiers.h File Reference

```
#include "symboles.h"
```

Functions

- void initialise_premiers (void)
- int est_premier (int non_terminal, int terminal)

Variables

- int premiers [NB_NON_TERMINAUX+1][NB_TERMINAUX+1]

4.7.1 Function Documentation

4.7.1.1 int est_premier (int *non_terminal,* int *terminal*)

4.7.1.2 void initialise_premiers (void)

4.7.2 Variable Documentation

4.7.2.1 int premiers[NB_NON_TERMINAUX+1][NB_TERMINAUX+1]

4.8 inc/suivants.h File Reference

```
#include "symboles.h"
```

Functions

- void initialise_suivants (void)
- int est_suivant (int non_terminal, int terminal)

Variables

- int suivants [NB_NON_TERMINAUX+1][NB_TERMINAUX+1]

4.8.1 Function Documentation

4.8.1.1 int est_suivant (int *non_terminal,* int *terminal*)

4.8.1.2 void initialise_suivants (void)

4.8.2 Variable Documentation

4.8.2.1 int suivants[NB_NON_TERMINAUX+1][NB_TERMINAUX+1]

4.9 inc/symboles.h File Reference

Macros

- #define EPSILON 0
- #define NB_NON_TERMINAUX 43
- #define _listeDecVariables_ 1
- #define _listeDecFonctions_ 2
- #define _declarationVariable_ 3
- #define _declarationFonction_ 4
- #define _listeParam_ 5
- #define _listeInstructions_ 6
- #define _instruction_ 8
- #define _instructionAffect_ 9
- #define _instructionBloc_ 10
- #define _instructionSi_ 11
- #define _instructionTantque_ 12
- #define _instructionAppel_ 13
- #define _instructionRetour_ 14
- #define _instructionEcriture_ 15
- #define _instructionVide_ 16
- #define _var_ 17
- #define _expression_ 18
- #define _appelFct_ 19
- #define _conjonction_ 20
- #define _negation_ 21
- #define _comparaison_ 22
- #define _expArith_ 23
- #define _terme_ 24
- #define _facteur_ 25

- #define _argumentsEffectifs_ 26
- #define _listeExpressions_ 27
- #define _listeExpressionsBis_ 7
- #define _programme_ 28
- #define _conjonctionBis_ 29
- #define _optTailleTableau_ 30
- #define _expArithBis_ 31
- #define _optSinon_ 32
- #define _comparaisonBis_ 33
- #define _optDecVariables_ 34
- #define _optIndice_ 35
- #define _listeDecVariablesBis_ 36
- #define _termeBis_ 38
- #define _expressionBis_ 39
- #define _instructionFaire_ 40
- #define _optListeDecVariables_ 41
- #define _instructionPour_ 42
- #define NB_TERMINAUX 32
- #define POINT_VIRGULE 1
- #define PLUS 2
- #define MOINS 3
- #define FOIS 4
- #define DIVISE 5
- #define PARENTHESE_OUVRANTE 6
- #define PARENTHESE_FERMANTE 7
- #define CROCHET_OUVRANT 8
- #define CROCHET_FERMANT 9
- #define ACCOLADE_OUVRANTE 10
- #define ACCOLADE_FERMANTE 11
- #define EGAL 12
- #define INFERIEUR 13
- #define ET 14
- #define OU 15
- #define NON 16
- #define SI 17
- #define ALORS 18
- #define SINON 19
- #define TANTQUE 20
- #define FAIRE 21
- #define ENTIER 22
- #define RETOUR 23
- #define LIRE 24
- #define ECRIRE 25
- #define ID_VAR 26
- #define ID_FCT 27
- #define NOMBRE 28
- #define FIN 29
- #define VIRGULE 30
- #define POUR 31

4.9.1 Macro Definition Documentation

4.9.1.1 #define _appelFct_ 19

4.9.1.2 #define _argumentsEffectifs_ 26

4.9.1.3 #define _comparaison_ 22

4.9.1.4 #define _comparaisonBis_ 33

4.9.1.5 #define _conjonction_ 20

4.9.1.6 #define _conjonctionBis_ 29

4.9.1.7 #define _declarationFonction_ 4

4.9.1.8 #define _declarationVariable_ 3

4.9.1.9 #define _expArith_ 23

4.9.1.10 #define _expArithBis_ 31

4.9.1.11 #define _expression_ 18

4.9.1.12 #define _expressionBis_ 39

4.9.1.13 #define _facteur_ 25

4.9.1.14 #define _instruction_ 8

4.9.1.15 #define _instructionAffect_ 9

4.9.1.16 #define _instructionAppel_ 13

4.9.1.17 #define _instructionBloc_ 10

4.9.1.18 #define _instructionEcriture_ 15

4.9.1.19 #define _instructionFaire_ 40

4.9.1.20 #define _instructionPour_ 42

4.9.1.21 #define _instructionRetour_ 14

4.9.1.22 #define _instructionSi_ 11

4.9.1.23 #define _instructionTantque_ 12

4.9.1.24 #define _instructionVide_ 16

4.9.1.25 #define _listeDecFonctions_ 2

4.9.1.26 #define _listeDecVariables_ 1

4.9.1.27 #define _listeDecVariablesBis_ 36

4.9.1.28 #define _listeExpressions_ 27

4.9.1.29 #define _listeExpressionsBis_ 7

4.9.1.30 #define _listeInstructions_ 6

4.9.1.31 #define _listeParam_ 5

4.9.1.32 #define _negation_ 21

4.9.1.33 #define _optDecVariables_ 34

4.9.1.34 #define _optIndice_ 35

4.9.1.35 #define _optListeDecVariables_ 41

4.9.1.36 #define _optSinon_ 32

4.9.1.37 #define _optTailleTableau_ 30

4.9.1.38 #define _programme_ 28

4.9.1.39 #define _terme_ 24

4.9.1.40 #define _termeBis_ 38

4.9.1.41 #define _var_ 17

4.9.1.42 #define ACCOLADE_FERMANTE 11

4.9.1.43 #define ACCOLADE_OUVRANTE 10

4.9.1.44 #define ALORS 18

4.9.1.45 #define CROCHET_FERMANT 9

4.9.1.46 #define CROCHET_OUVRANT 8

4.9.1.47 #define DIVISE 5

4.9.1.48 #define ECRIRE 25

4.9.1.49 #define EGAL 12

4.9.1.50 #define ENTIER 22

4.9.1.51 #define EPSILON 0

4.9.1.52 #define ET 14

4.9.1.53 #define FAIRE 21

4.9.1.54 #define FIN 29

4.9.1.55 #define FOIS 4

4.9.1.56 #define ID_FCT 27

4.9.1.57 #define ID_VAR 26

4.9.1.58 #define INFERIEUR 13

4.9.1.59 #define LIRE 24

4.9.1.60 #define MOINS 3

4.9.1.61 #define NB_NON_TERMINAUX 43

4.9.1.62 #define NB_TERMINAUX 32

4.9.1.63 #define NOMBRE 28

4.9.1.64 #define NON 16

4.9.1.65 #define OU 15

4.9.1.66 #define PARENTHESE_FERMANTE 7

4.9.1.67 #define PARENTHESE_OUVRANTE 6

4.9.1.68 #define PLUS 2

4.9.1.69 #define POINT_VIRGULE 1

4.9.1.70 #define POUR 31

4.9.1.71 #define RETOUR 23

4.9.1.72 #define SI 17

4.9.1.73 #define SINON 19

4.9.1.74 #define TANTQUE 20

4.9.1.75 #define VIRGULE 30

4.10 inc/syntabs.h File Reference

Data Structures

- struct n_prog_
- struct n_dec_
- struct n_exp_
- struct n_instr_
- struct n_appel_
- struct n_var_
- struct n_l_exp_
- struct n_l_instr_
- struct n_l_dec_

Typedefs

- typedef struct n_l_instr_ n_l_instr
- typedef struct n_instr_ n_instr
- typedef struct n_exp_ n_exp
- typedef struct n_l_exp_ n_l_exp
- typedef struct n_var_ n_var
- typedef struct n_l_dec_ n_l_dec
- typedef struct n_dec_ n_dec
- typedef struct n_prog_ n_prog
- typedef struct n_appel_ n_appel

Enumerations

- enum operation {
  plus, moins, fois, divise,
  modulo, egal, diff, inf,
  sup, infeg, supeg, ou,
  et, non, negatif }

Functions

- n_prog * cree_n_prog (n_l_dec *variables, n_l_dec *fonctions)
- n_dec * cree_n_dec_var (char *nom)
- n_dec * cree_n_dec_tab (char *nom, int taille)
- n_dec * cree_n_dec_fonc (char *nom, n_l_dec *param, n_l_dec *variables, n_instr *corps)
- n_exp * cree_n_exp_op (operation type, n_exp *op1, n_exp *op2)
- n_exp * cree_n_exp_entier (int entier)
- n_exp * cree_n_exp_var (n_var *var)
- n_exp * cree_n_exp_appel (n_appel *app)
- n_exp * cree_n_exp_lire (void)
- n_exp * cree_n_exp_incr (n_var *var)
- n_instr * cree_n_instr_incr (n_exp *incr)
- n_instr * cree_n_instr_si (n_exp *test, n_instr *alors, n_instr *sinon)
- n_instr * cree_n_instr_bloc (n_l_instr *liste)
- n_instr * cree_n_instr_tantque (n_exp *test, n_instr *faire)
- n_instr * cree_n_instr_faire (n_instr *faire, n_exp *test)
- n_instr * cree_n_instr_pour (n_instr *init, n_exp *test, n_instr *incr, n_instr *faire)
- n_instr * cree_n_instr_affect (n_var *var, n_exp *exp)
- n_instr * cree_n_instr_appel (n_appel *appel)
- n_instr * cree_n_instr_retour (n_exp *expression)
- n_instr * cree_n_instr_ecrire (n_exp *expression)
- n_instr * cree_n_instr_vide (void)
- n_appel * cree_n_appel (char *fonction, n_l_exp *args)
- n_var * cree_n_var_simple (char *nom)
- n_var * cree_n_var_indicee (char *nom, n_exp *indice)
- n_l_exp * cree_n_l_exp (n_exp *tete, n_l_exp *queue)
- n_l_instr * cree_n_l_instr (n_instr *tete, n_l_instr *queue)
- n_l_dec * cree_n_l_dec (n_dec *tete, n_l_dec *queue)

4.10.1 Typedef Documentation

4.10.1.1 typedef struct n_appel_ n_appel

4.10.1.2 typedef struct n_dec_ n_dec

4.10.1.3 typedef struct n_exp_ n_exp

4.10.1.4 typedef struct n_instr_ n_instr

4.10.1.5 typedef struct n_l_dec_ n_l_dec

4.10.1.6 typedef struct n_l_exp_ n_l_exp

4.10.1.7 typedef struct n_l_instr_ n_l_instr

4.10.1.8 typedef struct n_prog_ n_prog

4.10.1.9 typedef struct n_var_ n_var

4.10.2 Enumeration Type Documentation

4.10.2.1 enum operation

Enumerator

plus

moins
fois
divise
modulo
egal
diff
inf
sup
infeg
supeg
ou
et
non
negatif

4.10.3 Function Documentation

4.10.3.1 **n_appel**∗ cree_n_appel (char ∗ *fonction,* **n_l_exp** ∗ *args*)

4.10.3.2 **n_dec**∗ cree_n_dec_fonc (char ∗ *nom,* **n_l_dec** ∗ *param,* **n_l_dec** ∗ *variables,* **n_instr** ∗ *corps*)

4.10.3.3 **n_dec**∗ cree_n_dec_tab (char ∗ *nom,* int *taille*)

4.10.3.4 **n_dec**∗ cree_n_dec_var (char ∗ *nom*)

4.10.3.5 **n_exp**∗ cree_n_exp_appel (**n_appel** ∗ *app*)

4.10.3.6 **n_exp**∗ cree_n_exp_entier (int *entier*)

4.10.3.7 **n_exp**∗ cree_n_exp_incr (**n_var** ∗ *var*)

4.10.3.8 **n_exp**∗ cree_n_exp_lire (void)

4.10.3.9 **n_exp**∗ cree_n_exp_op (**operation** *type,* **n_exp** ∗ *op1,* **n_exp** ∗ *op2*)

4.10.3.10 **n_exp**∗ cree_n_exp_var (**n_var** ∗ *var*)

4.10.3.11 **n_instr**∗ cree_n_instr_affect (**n_var** ∗ *var,* **n_exp** ∗ *exp*)

4.10.3.12 **n_instr**∗ cree_n_instr_appel (**n_appel** ∗ *appel*)

4.10.3.13 **n_instr**∗ cree_n_instr_bloc (**n_l_instr** ∗ *liste*)

4.10.3.14 **n_instr**∗ cree_n_instr_ecrire (**n_exp** ∗ *expression*)

4.10.3.15 **n_instr*** cree_n_instr_faire (**n_instr** * *faire,* **n_exp** * *test*)

4.10.3.16 **n_instr*** cree_n_instr_incr (**n_exp** * *incr*)

4.10.3.17 **n_instr*** cree_n_instr_pour (**n_instr** * *init,* **n_exp** * *test,* **n_instr** * *incr,* **n_instr** * *faire*)

4.10.3.18 **n_instr*** cree_n_instr_retour (**n_exp** * *expression*)

4.10.3.19 **n_instr*** cree_n_instr_si (**n_exp** * *test,* **n_instr** * *alors,* **n_instr** * *sinon*)

4.10.3.20 **n_instr*** cree_n_instr_tantque (**n_exp** * *test,* **n_instr** * *faire*)

4.10.3.21 **n_instr*** cree_n_instr_vide (void)

4.10.3.22 **n_l_dec*** cree_n_l_dec (**n_dec** * *tete,* **n_l_dec** * *queue*)

4.10.3.23 **n_l_exp*** cree_n_l_exp (**n_exp** * *tete,* **n_l_exp** * *queue*)

4.10.3.24 **n_l_instr*** cree_n_l_instr (**n_instr** * *tete,* **n_l_instr** * *queue*)

4.10.3.25 **n_prog*** cree_n_prog (**n_l_dec** * *variables,* **n_l_dec** * *fonctions*)

4.10.3.26 **n_var*** cree_n_var_indicee (char * *nom,* **n_exp** * *indice*)

4.10.3.27 **n_var*** cree_n_var_simple (char * *nom*)

4.11 inc/util.h File Reference

Functions

- char * duplique_chaine (char *s)
- void erreur (char *message)
- void erreur_1s (char *message, char *s)
- void affiche_balise_ouvrante (const char *fct_, int trace_xml)
- void affiche_balise_fermante (const char *fct_, int trace_xml)
- void affiche_element (char *fct_, char *texte_, int trace_xml)
- void affiche_texte (char *texte_, int trace_xml)

4.11.1 Function Documentation

4.11.1.1 void affiche_balise_fermante (const char * *fct_*, int *trace_xml*)

4.11.1.2 void affiche_balise_ouvrante (const char * *fct_*, int *trace_xml*)

4.11.1.3 void affiche_element (char * *fct_*, char * *texte_*, int *trace_xml*)

4.11.1.4 void affiche_texte (char * *texte_*, int *trace_xml*)

4.11.1.5 char* duplique_chaine (char * *s*)

4.11.1.6 void erreur (char * *message*)

4.11.1.7 void erreur_1s (char * *message*, char * *s*)

4.12 src/affiche_arbre_abstrait.c File Reference

```
#include <stdio.h>
#include "syntabs.h"
#include "util.h"
```

Functions

- void affiche_n_prog (n_prog *n)
- void affiche_l_instr (n_l_instr *n)
- void affiche_instr (n_instr *n)
- void affiche_instr_si (n_instr *n)
- void affiche_instr_tantque (n_instr *n)
- void affiche_instr_faire (n_instr *n)
- void affiche_instr_pour (n_instr *n)
- void affiche_instr_affect (n_instr *n)
- void affiche_instr_appel (n_instr *n)
- void affiche_instr_retour (n_instr *n)
- void affiche_instr_ecrire (n_instr *n)
- void affiche_l_exp (n_l_exp *n)
- void affiche_exp (n_exp *n)
- void affiche_varExp (n_exp *n)
- void affiche_opExp (n_exp *n)
- void affiche_intExp (n_exp *n)
- void affiche_lireExp (n_exp *n)
- void affiche_appelExp (n_exp *n)
- void affiche_l_dec (n_l_dec *n)
- void affiche_dec (n_dec *n)
- void affiche_foncDec (n_dec *n)
- void affiche_varDec (n_dec *n)
- void affiche_tabDec (n_dec *n)
- void affiche_var (n_var *n)
- void affiche_var_simple (n_var *n)
- void affiche_var_indicee (n_var *n)
- void affiche_appel (n_appel *n)

Variables

- int trace_abs = 1

4.12.1 Function Documentation

4.12.1.1 void affiche_appel (n_appel * n)

4.12.1.2 void affiche_appelExp (n_exp * n)

4.12.1.3 void affiche_dec (n_dec * n)

4.12.1.4 void affiche_exp (n_exp * n)

4.12.1.5 void affiche_foncDec (n_dec * n)

4.12.1.6 void affiche_instr (n_instr * n)

4.12.1.7 void affiche_instr_affect (n_instr * n)

4.12.1.8 void affiche_instr_appel (n_instr * n)

4.12.1.9 void affiche_instr_ecrire (n_instr * n)

4.12.1.10 void affiche_instr_faire (n_instr * n)

4.12.1.11 void affiche_instr_pour (n_instr * n)

4.12.1.12 void affiche_instr_retour (n_instr * n)

4.12.1.13 void affiche_instr_si (n_instr * n)

4.12.1.14 void affiche_instr_tantque (n_instr * n)

4.12.1.15 void affiche_intExp (n_exp * n)

4.12.1.16 void affiche_l_dec (n_l_dec * n)

4.12.1.17 void affiche_l_exp (n_l_exp * n)

4.12.1.18 void affiche_l_instr (n_l_instr * n)

4.12.1.19 void affiche_lireExp (n_exp * n)

4.12.1.20 void affiche_n_prog (n_prog * n)

4.12.1.21 void affiche_opExp (n_exp * n)

4.12.1.22 void affiche_tabDec (n_dec * n)

4.12.1.23 void affiche_var (n_var * n)

4.12.1.24 void affiche_var_indicee (n_var * n)

4.12.1.25 void affiche_var_simple (n_var * n)

4.12.1.26 void affiche_varDec (n_dec * n)

4.12.1.27 void affiche_varExp (n_exp * n)

4.12.2 Variable Documentation

4.12.2.1 int trace_abs = 1

4.13 src/analyseur_lexical.c File Reference

```
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include "symboles.h"
#include "analyseur_lexical.h"
#include "util.h"
```

Macros

- #define YYTEXT_MAX 100
- #define is_num(c) (('0' <= (c)) && ((c) <= '9'))
- #define is_maj(c) (('A' <= (c)) && ((c) <= 'Z'))
- #define is_min(c) (('a' <= (c)) && ((c) <= 'z'))
- #define is_alpha(c) (is_maj(c) || is_min(c) || (c) == '_' || (c) == '$')
- #define is_alphanum(c) (is_num((c)) || is_alpha((c)))

Functions

- int mangeEspaces ()
- char lireCar (void)
- void delireCar ()
- int yylex (void)
- void nom_token (int token, char *nom, char *valeur)
- void test_yylex_internal (FILE *yyin)

Variables

- FILE * yyin
- char * tableMotsClefs [ ]
- int codeMotClefs [ ]
- char yytext [YYTEXT_MAX]
- int yyleng
- int nbMotsClefs = 10
- int motsClefsMaxLeng = 7
- int nb_ligne = 1

4.13.1 Macro Definition Documentation

4.13.1.1 #define is_alpha(*c*) (is_maj(c) || is_min(c) || (c) == '_' || (c) == '$')

4.13.1.2 #define is_alphanum(*c*) (is_num((c)) || is_alpha((c)))

4.13.1.3 #define is_maj(*c*) (('A' <= (c)) && ((c) <= 'Z'))

4.13.1.4 #define is_min(*c*) (('a' <= (c)) && ((c) <= 'z'))

4.13.1.5 #define is_num(*c*) (('0' <= (c)) && ((c) <= '9'))

4.13.1.6 #define YYTEXT_MAX 100

4.13.2 Function Documentation

4.13.2.1 void delireCar ( )

4.13.2.2 char lireCar (void)

4.13.2.3 int mangeEspaces ( )

4.13.2.4 void nom_token (int *token*, char * *nom*, char * *valeur*)

4.13.2.5 void test_yylex_internal (FILE * *yyin*)

4.13.2.6 int yylex (void)

4.13.3 Variable Documentation

4.13.3.1 int codeMotClefs[ ]

Initial value:

```
= {
  SI, ALORS, SINON, TANTQUE, FAIRE, RETOUR, ENTIER,
      LIRE, ECRIRE, POUR
}
```

4.13.3.2 int motsClefsMaxLeng = 7

4.13.3.3 int nb_ligne = 1

4.13.3.4 int nbMotsClefs = 10

4.13.3.5 char* tableMotsClefs[ ]

Initial value:

```
= {
  "si", "alors", "sinon", "tantque", "faire", "retour", "entier", "lire", "ecrire" , "pour"
}
```

4.13.3.6 FILE* yyin

4.13.3.7 int yyleng

4.13.3.8 char yytext[YYTEXT_MAX]

4.14 src/analyseur_semantique.c File Reference

```
#include <stdarg.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "analyseur_semantique.h"
#include "syntabs.h"
#include "util.h"
#include "dico.h"
```

Functions

- void analyse_n_prog (n_prog *n)
- void analyse_l_instr (n_l_instr *n)
- void analyse_instr (n_instr *n)
- void analyse_instr_si (n_instr *n)
- void analyse_instr_tantque (n_instr *n)
- void analyse_instr_faire (n_instr *n)
- void analyse_instr_pour (n_instr *n)
- void analyse_instr_affect (n_instr *n)
- void analyse_instr_appel (n_instr *n)
- void analyse_instr_retour (n_instr *n)
- void analyse_instr_ecrire (n_instr *n)
- void analyse_l_exp (n_l_exp *n)
- void analyse_exp (n_exp *n)
- void analyse_varExp (n_exp *n)
- void analyse_opExp (n_exp *n)

- void analyse_intExp (n_exp *n)
- void analyse_lireExp (n_exp *n)
- void analyse_appelExp (n_exp *n)
- void analyse_l_dec (n_l_dec *n)
- void analyse_dec (n_dec *n)
- void analyse_foncDec (n_dec *n)
- void analyse_varDec (n_dec *n)
- void analyse_tabDec (n_dec *n)
- void analyse_var (n_var *n, char *s)
- void analyse_var_simple (n_var *n, char *s)
- void analyse_var_indicee (n_var *n, char *s)
- void analyse_appel (n_appel *n)
- void semantique (n_prog *p, int trace_dico, int print_mips)

  Main function to parse syntaxique tree.
- int taille_n_l_dec (n_l_dec *liste)
- int taille_n_l_exp (n_l_exp *liste)
- int newEtiquette (void)
- int mips_print (const char *format,...)
- void mips_debut_fonction ()
- void mips_fin_function ()
- void mips_empile (char *s)
- void mips_depile (char *s)

Variables

- int contexte = C_VARIABLE_GLOBALE
- int adresseLocaleCourante = 0
- int adresseArgumentCourant = 0
- int trace_tab
- int trace_mips
- int nb_args_function
- int nb_var_local
- int etiquette = 0
- int asreturn

4.14.1 Function Documentation

4.14.1.1 void analyse_appel (n_appel * *n*)

4.14.1.2 void analyse_appelExp (n_exp * *n*)

4.14.1.3 void analyse_dec (n_dec * *n*)

4.14.1.4 void analyse_exp (n_exp * *n*)

4.14.1.5 void analyse_foncDec (n_dec * *n*)

4.14.1.6 void analyse_instr (n_instr * *n*)

4.14.1.7 void analyse_instr_affect (n_instr * n)

4.14.1.8 void analyse_instr_appel (n_instr * n)

4.14.1.9 void analyse_instr_ecrire (n_instr * n)

4.14.1.10 void analyse_instr_faire (n_instr * n)

4.14.1.11 void analyse_instr_pour (n_instr * n)

4.14.1.12 void analyse_instr_retour (n_instr * n)

4.14.1.13 void analyse_instr_si (n_instr * n)

4.14.1.14 void analyse_instr_tantque (n_instr * n)

4.14.1.15 void analyse_intExp (n_exp * n)

4.14.1.16 void analyse_l_dec (n_l_dec * n)

4.14.1.17 void analyse_l_exp (n_l_exp * n)

4.14.1.18 void analyse_l_instr (n_l_instr * n)

4.14.1.19 void analyse_lireExp (n_exp * n)

4.14.1.20 void analyse_n_prog (n_prog * n)

4.14.1.21 void analyse_opExp (n_exp * n)

4.14.1.22 void analyse_tabDec (n_dec * n)

4.14.1.23 void analyse_var (n_var * n, char * s)

4.14.1.24 void analyse_var_indicee (n_var * n, char * s)

4.14.1.25 void analyse_var_simple (n_var * n, char * s)

4.14.1.26 void analyse_varDec (n_dec * n)

4.14.1.27 void analyse_varExp (n_exp * n)

4.14.1.28 void mips_debut_fonction ( )

4.14.1.29 void mips_depile (char * s)

4.14.1.30 void mips_empile (char * s)

4.14.1.31 void mips_fin_function ( )

4.14.1.32 int mips_print (const char * format, ...)

4.14.1.33 int newEtiquette (void)

4.14.1.34 void semantique (n_prog * p, int trace_dico, int print_mips)

Main function to parse syntaxique tree.

Parameters

	p	The head of the tree
in	*trace_dico*	boolean to display symbol table
in	*print_mips*	boolean to display the mips output on stdin

4.14.1.35 int taille_n_l_dec (n_l_dec * *liste*)

4.14.1.36 int taille_n_l_exp (n_l_exp * *liste*)

4.14.2 Variable Documentation

4.14.2.1 int adresseArgumentCourant = 0

4.14.2.2 int adresseLocaleCourante = 0

4.14.2.3 int asreturn

4.14.2.4 int contexte = C_VARIABLE_GLOBALE

4.14.2.5 int etiquette = 0

4.14.2.6 int nb_args_function

4.14.2.7 int nb_var_local

4.14.2.8 int trace_mips

4.14.2.9 int trace_tab

4.15 src/analyseur_syntaxique.c File Reference

```
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include "symboles.h"
#include "analyseur_syntaxique.h"
#include "util.h"
#include "analyseur_lexical.h"
#include "premiers.h"
#include "suivants.h"
```

Functions

- void EatTerminal (void)
- void DisplayErreur (void)
- n_l_dec * listeDecVariables (void)
- n_l_dec * listeDecFonctions (void)
- n_dec * declarationVariable (void)
- n_dec * declarationFonction (void)
- n_l_dec * listeParam (void)
- n_l_instr * listeInstructions (void)
- n_instr * instruction (void)
- n_instr * instructionAffect (void)
- n_instr * instructionBloc (void)
- n_instr * instructionSi (void)
- n_instr * instructionTantque (void)
- n_instr * instructionAppel (void)
- n_instr * instructionRetour (void)
- n_instr * instructionEcriture (void)
- n_instr * instructionVide (void)
- n_var * var (void)
- n_instr * instructionPour (void)
- n_exp * expression (void)
- n_appel * appelFct (void)
- n_exp * conjonction (void)
- n_exp * negation (void)
- n_exp * comparaison (void)
- n_exp * expArith (void)
- n_exp * terme (void)
- n_exp * facteur (void)
- n_l_exp * listeExpressions (void)
- n_l_exp * listeExpressionsBis (n_l_exp *herite)
- n_prog * programme (void)
- n_exp * conjonctionBis (n_exp *herite)
- int optTailleTableau (void)
- n_exp * expArithBis (n_exp *herite)
- n_instr * optSinon (void)
- n_exp * comparaisonBis (n_exp *herite)
- n_l_dec * optDecVariables (void)
- n_exp * optIndice (void)
- n_l_dec * listeDecVariablesBis ()
- n_exp * termeBis (n_exp *herite)
- n_exp * expressionBis (n_exp *herite)
- n_l_dec * optListeDecVariables (void)
- n_prog * syntaxe (int trace_xml_tree)

Variables

- int uniteCourante
- int trace_xml
- char nom [100]
- char valeur [100]
- int nb_ligne
- char yytext [ ]

4.15.1 Function Documentation

4.15.1.1 n_appel* appelFct (void)

4.15.1.2 n_exp* comparaison (void)

4.15.1.3 n_exp* comparaisonBis (n_exp * *herite*)

4.15.1.4 n_exp* conjonction (void)

4.15.1.5 n_exp* conjonctionBis (n_exp * *herite*)

4.15.1.6 n_dec* declarationFonction (void)

4.15.1.7 n_dec* declarationVariable (void)

4.15.1.8 void DisplayErreur (void)

4.15.1.9 void EatTerminal (void)

4.15.1.10 n_exp* expArith (void)

4.15.1.11 n_exp* expArithBis (n_exp * *herite*)

4.15.1.12 n_exp* expression (void)

4.15.1.13 n_exp* expressionBis (n_exp * *herite*)

4.15.1.14 n_exp* facteur (void)

4.15.1.15 n_instr* instruction (void)

4.15.1.16 n_instr* instructionAffect (void)

4.15.1.17 n_instr* instructionAppel (void)

4.15.1.18 n_instr* instructionBloc (void)

4.15.1.19 n_instr* instructionEcriture (void)

4.15.1.20 n_instr* instructionPour (void)

4.15.1.21 n_instr* instructionRetour (void)

4.15.1.22 n_instr* instructionSi (void)

4.15.1.23 n_instr* instructionTantque (void)

4.15.1.24 n_instr* instructionVide (void)

4.15.1.25 n_l_dec* listeDecFonctions (void)

4.15.1.26 n_l_dec* listeDecVariables (void)

4.15.1.27 n_l_dec* listeDecVariablesBis ( )

4.15.1.28 n_l_exp* listeExpressions (void)

4.15.1.29 n_l_exp* listeExpressionsBis (n_l_exp * *herite*)

4.15.1.30 n_l_instr* listeInstructions (void)

4.15.1.31 n_l_dec* listeParam (void)

4.15.1.32 n_exp* negation (void)

4.15.1.33 n_l_dec* optDecVariables (void)

4.15.1.34 n_exp* optIndice (void)

4.15.1.35 n_l_dec* optListeDecVariables (void)

4.15.1.36 n_instr* optSinon (void)

4.15.1.37 int optTailleTableau (void)

4.15.1.38 n_prog* programme (void)

4.15.1.39 n_prog* syntaxe (int *trace_xml_tree*)

4.15.1.40 n_exp* terme (void)

4.15.1.41 n_exp* termeBis (n_exp * *herite*)

4.15.1.42 n_var* var (void)

4.15.2 Variable Documentation

4.15.2.1 int nb_ligne

4.15.2.2 char nom[100]

4.15.2.3 int trace_xml

4.15.2.4 int uniteCourante

4.15.2.5 char valeur[100]

4.15.2.6 char yytext[ ]

4.16 src/compilateur.c File Reference

```
#include <ctype.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include "affiche_arbre_abstrait.h"
#include "analyseur_lexical.h"
#include "analyseur_syntaxique.h"
#include "analyseur_semantique.h"
#include "freedom.h"
```

Functions

- int main (int argc, char **argv)

Variables

- char yytext [100]
- FILE * yyin

4.16.1 Function Documentation

4.16.1.1 int main (int *argc,* char ** *argv*)

4.16.2 Variable Documentation

4.16.2.1 FILE* yyin

4.16.2.2 char yytext[100]

4.17 src/dico.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "dico.h"
```

Functions

- void entreeFonction (void)
- void sortieFonction (void)
- int ajouteIdentificateur (char *identif, int classe, int type, int adresse, int complement)
- int rechercheExecutable (char *identif)
- int rechercheDeclarative (char *identif)
- void affiche_dico (void)

4.17.1 Function Documentation

4.17.1.1 void affiche_dico (void)

4.17.1.2 int ajouteIdentificateur (char * *identif,* int *classe,* int *type,* int *adresse,* int *complement*)

4.17.1.3 void entreeFonction (void)

4.17.1.4 int rechercheDeclarative (char * *identif*)

4.17.1.5 int rechercheExecutable (char * *identif*)

4.17.1.6 void sortieFonction (void)

4.18 src/freedom.c File Reference

```
#include <stdlib.h>
#include "syntabs.h"
#include "dico.h"
```

Functions

- void free_dico ()
- void free_n_prog (n_prog *n)
- void free_l_instr (n_l_instr *n)
- void free_instr (n_instr *n)
- void free_instr_si (n_instr *n)
- void free_instr_tantque (n_instr *n)
- void free_instr_faire (n_instr *n)
- void free_instr_pour (n_instr *n)
- void free_instr_affect (n_instr *n)
- void free_instr_appel (n_instr *n)
- void free_instr_retour (n_instr *n)
- void free_instr_ecrire (n_instr *n)
- void free_l_exp (n_l_exp *n)
- void free_exp (n_exp *n)
- void free_varExp (n_exp *n)
- void free_opExp (n_exp *n)
- void free_intExp (n_exp *n)

- void free_lireExp (n_exp ∗n)
- void free_appelExp (n_exp ∗n)
- void free_l_dec (n_l_dec ∗n)
- void free_dec (n_dec ∗n)
- void free_foncDec (n_dec ∗n)
- void free_varDec (n_dec ∗n)
- void free_tabDec (n_dec ∗n)
- void free_var (n_var ∗n)
- void free_var_simple (n_var ∗n)
- void free_var_indicee (n_var ∗n)
- void free_appel (n_appel ∗n)
- void freedom (n_prog ∗n)

4.18.1 Function Documentation

4.18.1.1 void free_appel (n_appel ∗ *n*)

4.18.1.2 void free_appelExp (n_exp ∗ *n*)

4.18.1.3 void free_dec (n_dec ∗ *n*)

4.18.1.4 void free_dico ( )

4.18.1.5 void free_exp (n_exp ∗ *n*)

4.18.1.6 void free_foncDec (n_dec ∗ *n*)

4.18.1.7 void free_instr (n_instr ∗ *n*)

4.18.1.8 void free_instr_affect (n_instr ∗ *n*)

4.18.1.9 void free_instr_appel (n_instr ∗ *n*)

4.18.1.10 void free_instr_ecrire (n_instr ∗ *n*)

4.18.1.11 void free_instr_faire (n_instr ∗ *n*)

4.18.1.12 void free_instr_pour (n_instr ∗ *n*)

4.18.1.13 void free_instr_retour (n_instr ∗ *n*)

4.18.1.14 void free_instr_si (n_instr ∗ *n*)

4.18.1.15 void free_instr_tantque (n_instr ∗ *n*)

4.18.1.16 void free_intExp (n_exp ∗ *n*)

4.18.1.17 void free_l_dec (n_l_dec * n)

4.18.1.18 void free_l_exp (n_l_exp * n)

4.18.1.19 void free_l_instr (n_l_instr * n)

4.18.1.20 void free_lireExp (n_exp * n)

4.18.1.21 void free_n_prog (n_prog * n)

4.18.1.22 void free_opExp (n_exp * n)

4.18.1.23 void free_tabDec (n_dec * n)

4.18.1.24 void free_var (n_var * n)

4.18.1.25 void free_var_indicee (n_var * n)

4.18.1.26 void free_var_simple (n_var * n)

4.18.1.27 void free_varDec (n_dec * n)

4.18.1.28 void free_varExp (n_exp * n)

4.18.1.29 void freedom (n_prog * n)

4.19 src/premiers.c File Reference

```
#include "symboles.h"
#include "premiers.h"
```

Functions

- void initialise_premiers (void)
- int est_premier (int terminal, int non_terminal)

4.19.1 Function Documentation

4.19.1.1 int est_premier (int *terminal*, int *non_terminal*)

4.19.1.2 void initialise_premiers (void)

4.20 src/suivants.c File Reference

```
#include "symboles.h"
#include "suivants.h"
```

Functions

- void initialise_suivants (void)
- int est_suivant (int terminal, int non_terminal)

4.20.1 Function Documentation

4.20.1.1 int est_suivant (int *terminal,* int *non_terminal*)

4.20.1.2 void initialise_suivants (void)

4.21 src/syntabs.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include "util.h"
#include "syntabs.h"
```

Functions

- n_appel * cree_n_appel (char *fonction, n_l_exp *args)
- n_prog * cree_n_prog (n_l_dec *variables, n_l_dec *fonctions)
- n_var * cree_n_var_simple (char *nom)
- n_var * cree_n_var_indicee (char *nom, n_exp *indice)
- n_exp * cree_n_exp_op (operation op, n_exp *op1, n_exp *op2)
- n_exp * cree_n_exp_appel (n_appel *app)
- n_exp * cree_n_exp_incr (n_var *var)
- n_exp * cree_n_exp_var (n_var *var)
- n_exp * cree_n_exp_entier (int entier)
- n_exp * cree_n_exp_lire ()
- n_l_exp * cree_n_l_exp (n_exp *tete, n_l_exp *queue)
- n_instr * cree_n_instr_incr (n_exp *incr)
- n_instr * cree_n_instr_si (n_exp *test, n_instr *alors, n_instr *sinon)
- n_instr * cree_n_instr_tantque (n_exp *test, n_instr *faire)
- n_instr * cree_n_instr_faire (n_instr *faire, n_exp *test)
- n_instr * cree_n_instr_pour (n_instr *init, n_exp *test, n_instr *incr, n_instr *faire)
- n_instr * cree_n_instr_affect (n_var *var, n_exp *exp)
- n_l_instr * cree_n_l_instr (n_instr *tete, n_l_instr *queue)
- n_instr * cree_n_instr_bloc (n_l_instr *liste)
- n_instr * cree_n_instr_appel (n_appel *app)
- n_instr * cree_n_instr_ecrire (n_exp *expression)
- n_instr * cree_n_instr_retour (n_exp *expression)
- n_instr * cree_n_instr_vide (void)
- n_dec * cree_n_dec_var (char *nom)
- n_dec * cree_n_dec_tab (char *nom, int taille)
- n_dec * cree_n_dec_fonc (char *nom, n_l_dec *param, n_l_dec *variables, n_instr *corps)
- n_l_dec * cree_n_l_dec (n_dec *tete, n_l_dec *queue)

4.21.1 Function Documentation

4.21.1.1 **n_appel**∗ cree_n_appel (char ∗ *fonction,* **n_l_exp** ∗ *args*)

4.21.1.2 **n_dec**∗ cree_n_dec_fonc (char ∗ *nom,* **n_l_dec** ∗ *param,* **n_l_dec** ∗ *variables,* **n_instr** ∗ *corps*)

4.21.1.3 **n_dec**∗ cree_n_dec_tab (char ∗ *nom,* int *taille*)

4.21.1.4 **n_dec**∗ cree_n_dec_var (char ∗ *nom*)

4.21.1.5 **n_exp**∗ cree_n_exp_appel (**n_appel** ∗ *app*)

4.21.1.6 **n_exp**∗ cree_n_exp_entier (int *entier*)

4.21.1.7 **n_exp**∗ cree_n_exp_incr (**n_var** ∗ *var*)

4.21.1.8 **n_exp**∗ cree_n_exp_lire (void)

4.21.1.9 **n_exp**∗ cree_n_exp_op (**operation** *op,* **n_exp** ∗ *op1,* **n_exp** ∗ *op2*)

4.21.1.10 **n_exp**∗ cree_n_exp_var (**n_var** ∗ *var*)

4.21.1.11 **n_instr**∗ cree_n_instr_affect (**n_var** ∗ *var,* **n_exp** ∗ *exp*)

4.21.1.12 **n_instr**∗ cree_n_instr_appel (**n_appel** ∗ *app*)

4.21.1.13 **n_instr**∗ cree_n_instr_bloc (**n_l_instr** ∗ *liste*)

4.21.1.14 **n_instr**∗ cree_n_instr_ecrire (**n_exp** ∗ *expression*)

4.21.1.15 **n_instr**∗ cree_n_instr_faire (**n_instr** ∗ *faire,* **n_exp** ∗ *test*)

4.21.1.16 **n_instr**∗ cree_n_instr_incr (**n_exp** ∗ *incr*)

4.21.1.17 **n_instr**∗ cree_n_instr_pour (**n_instr** ∗ *init,* **n_exp** ∗ *test,* **n_instr** ∗ *incr,* **n_instr** ∗ *faire*)

4.21.1.18 **n_instr**∗ cree_n_instr_retour (**n_exp** ∗ *expression*)

4.21.1.19 **n_instr**∗ cree_n_instr_si (**n_exp** ∗ *test,* **n_instr** ∗ *alors,* **n_instr** ∗ *sinon*)

4.21.1.20 **n_instr**∗ cree_n_instr_tantque (**n_exp** ∗ *test,* **n_instr** ∗ *faire*)

4.21.1.21 **n_instr**∗ cree_n_instr_vide (void)

4.21.1.22 **n_l_dec**∗ cree_n_l_dec (**n_dec** ∗ *tete,* **n_l_dec** ∗ *queue*)

4.21.1.23 n_l_exp* cree_n_l_exp (n_exp * *tete*, n_l_exp * *queue*)

4.21.1.24 n_l_instr* cree_n_l_instr (n_instr * *tete*, n_l_instr * *queue*)

4.21.1.25 n_prog* cree_n_prog (n_l_dec * *variables*, n_l_dec * *fonctions*)

4.21.1.26 n_var* cree_n_var_indicee (char * *nom*, n_exp * *indice*)

4.21.1.27 n_var* cree_n_var_simple (char * *nom*)

4.22 src/util.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Functions

- void erreur (char *message)
- void erreur_1s (char *message, char *s)
- char * duplique_chaine (char *src)
- void indent ()
- void affiche_balise_ouvrante (const char *fct_, int trace_xml)
- void affiche_balise_fermante (const char *fct_, int trace_xml)
- void affiche_texte (char *texte_, int trace_xml)
- void affiche_xml_texte (char *texte_)
- void affiche_element (char *fct_, char *texte_, int trace_xml)

Variables

- int nb_ligne
- int indent_xml = 0
- int indent_step = 1

4.22.1 Function Documentation

4.22.1.1 void affiche_balise_fermante (const char * *fct_*, int *trace_xml*)

4.22.1.2 void affiche_balise_ouvrante (const char * *fct_*, int *trace_xml*)

4.22.1.3 void affiche_element (char * *fct_*, char * *texte_*, int *trace_xml*)

4.22.1.4 void affiche_texte (char * *texte_*, int *trace_xml*)

4.22.1.5 void affiche_xml_texte (char * *texte_*)

4.22.1.6 char* duplique_chaine (char * *src*)

4.22.1.7 void erreur (char * *message*)

4.22.1.8 void erreur_1s (char * *message,* char * *s*)

4.22.1.9 void indent ( )

4.22.2 Variable Documentation

4.22.2.1 int indent_step = 1

4.22.2.2 int indent_xml = 0

4.22.2.3 int nb_ligne

Index